1.
Napisati funkciju koji odreduje dva najveca elementa niza a i program koji je testira. Elementi niza se unose sa standardnog ulaza sve dok se ne unese 0. Pretpostaviti da niz nece imati vise od 32 elemenata.
Primer 1
ulaz 1 2 3 4 5 6 0
izlaz 6 5
Primer 2
ulaz: 3 4 -1 2 3 0
izlaz 4 3
Primer 3
ulaz 4 5 4 5 2 0
izlaz 5 5

RESENJE 1: http://bubblebee.rs/problemi/Zbirka/prvi_i_drugi_maksimum

Zadatak efikasnije možemo rešavati klasičnim postupkom određivanja maksimalnog elementa serije

Maksimalni element postavimo na prvi element serije (0-ti clan niza), a zatim jedan po jedan element serije upoređujemo sa tekućom vrednosti maksimuma.

- Ako je element serije koji upoređujemo veći ili jednak od vrednost tekućeg maksimuma, drugi po veličini broj serije dobija vrednost tekućeg maksimuma, a tekući maksimum postaje element serije koji smo upravo upoređivali.

- Ako je element serije manji od vrednosti tekućeg maksimuma potrebno ga je uporediti sa drugim po veličini brojem serije.

U poslednjem slučaju nastupa problem, jer drugi po veličini broj serije na početku nije definisan. To možemo prevazići ako uvedemo logičku promenljivu koja prati situaciju da li je drugi po veličini broj serije postavljen, na početku je 0, a kad drugi po veličini broj serije dobije vrednost postaje 1. Dakle, u poslednjem slučaju, ako drugi po veličini broj nije postavljen on dobija vrednost elementa serije koji analiziramo, a ako je postavljen onda se upoređuje sa elementom serije i u skladu sa rezultatom poređenja menja mu se vrednost.

```
#include <stdio.h>
#include <stdlib.h>

void Obradi(int tekuciClanNiza, int *prviMax, int *drugiMax, int *postavljenDrugiMax) {
  if (tekuciClanNiza >= *prviMax) {
    *drugiMax = *prviMax;
    *postavljenDrugiMax = 1;  //true;
    *prviMax = tekuciClanNiza;
  } else if ( ((*postavljenDrugiMax==0) || tekuciClanNiza >= *drugiMax)) {
    *drugiMax = tekuciClanNiza;
    *postavljenDrugiMax = 1;  //true
  }
}


int main()
{
```

```
        int niz[32], broj, i=0, dimenzija;

        //ucitavanje niza do unosa 0
        while(1)
   {
      scanf("%d", &broj);
      if (broj==0) break;
      niz[i++]=broj;
   }

   dimenzija=i;

  int prviMax, drugiMax;
  int postavljenDrugiMax;

  prviMax = niz[0];
  postavljenDrugiMax = 0;  // false

   for(i=1;i<dimenzija;i++)
       Obradi(niz[i], &prviMax, &drugiMax, &postavljenDrugiMax);


   printf("%d", prviMax);
   printf(" %d\n", drugiMax);

   return 0;
}
```

RESENJE 2 – REGULA-FALSI

```
#include <stdio.h>

int main()
{
   int niz[32], broj, i=0, dimenzija;

        //ucitavanje niza do unosa 0
        do
   {
      scanf("%d", &broj);
      if (broj==0) break;
      niz[i++]=broj;
      if (i==32) break;
   }while(1);

   dimenzija=i;

  int prviMax, drugiMax;

  prviMax = niz[0];
  drugiMax = niz[1];
```

```
 //trampa vrednosti da obezbedimo da prviMax>=drugiMax
 if (prviMax<drugiMax)
 {
    int pom=drugiMax;
    drugiMax=prviMax;
    prviMax=pom;
 }
   //postavljenje vrednosti za prviMax i drugiMax prolaskom kroz podniz niz[2..n-1]
  for(i=2;i<dimenzija;i++)
    if (niz[i]>=prviMax)
    {
       drugiMax=prviMax;  //OVAJ DEO IMPLEMENTACIJE JE CEST IZVOR GRESKI
       prviMax=niz[i];
    }
    else if (niz[i]>drugiMax)
         drugiMax=niz[i];


  printf("%d", prviMax);
  printf(" %d\n", drugiMax);

   return 0;
}
```

RESENJE 3 -SKELET

```
#include <stdio.h>
#include <stdlib.h>


int main()
{
        int niz[32], broj, i=0, dimenzija;

        //ucitavanje niza do unosa 0
        while(1)
  {
     scanf("%d", &broj);
     if (broj==0) break;
     niz[i++]=broj;
  }


    ??????????????
        printf("Prva dva clana:");
  for(i = 0; i < 2; i++) printf("%d ", niz[i]);

  return 0;
}
```

RESENJE 3 – 1. nacin

```
#include <stdio.h>
#include <stdlib.h>

//uredi nerastuce
void uredi(int s[], int n);

int main()
{
        int niz[32], broj, i=0, dimenzija;

        //ucitavanje niza do unosa 0
        while(1)
   {
      scanf("%d", &broj);
      if (broj==0) break;
      niz[i++]=broj;
   }

   dimenzija=i;

   uredi(niz, dimenzija);

        printf("Prva dva clana:");
   for(i = 0; i < 2; i++) printf("%d ", niz[i]);

   return 0;
}

void uredi(int s[], int n){

        int i, j;
        int tmp;
        /* sortiramo... */
        for(i=0;i<n-1;i++)
                for(j=i+1;j<n;j++)
                        if(s[i]<s[j]){
                                tmp=s[i];
                                s[i]=s[j];
                                s[j]=tmp;
                        }

}
```

RESENJE 3 – 2.nacin

```
#include <stdio.h>
#include <stdlib.h>

/* funkcije za poredjenje qsortom*/
/* funkcija za poredjenje celih brojeva u rastucem redosledu*/
```

```
int icmp(const void *p1, const void *p2);

int main()
{
        int niz[32], broj, i=0, dimenzija;

        //ucitavanje niza do unosa 0
        while(1)
   {
      scanf("%d", &broj);
      if (broj==0) break;
      niz[i++]=broj;
   }

   dimenzija=i;
        /* sortiranje celih brojeva*/
   qsort( niz, dimenzija, sizeof(int), icmp);

        printf("Prva dva clana:");
   for(i = 0; i < 2; i++) printf("%d ", niz[i]);

   return 0;
}


// funkcija za poredjenje celih brojeva u rastucem redosledu
int icmp(const void *p1, const void *p2)
{
        return ( (*(int *)p2) - (*(int *)p1) );
}
```

2.

Napisati funkciju

int palindrom (char s[])

kojom se proverava da li je zadata niska s palindrom (za nisku kazemo da je palindrom ako se isto cita i sa leve i sa desne strane).

Na primer: niska anavolimilovana je palindrom

U glavnom programu ucitati nisku i testirati rad funkcije.

```
RESENJE 1 – skelet
#include <stdio.h>
#include <string.h>

#define MAX_DUZINA 20
/* funkcija palindrom vraca vrednost 1 ako je niska s palindrom, inace vraca
vrednost 0 */
int palindrom(char s[]);
int main(){

        char s[MAX_DUZINA+1];
                printf("Unesite nisku: ");
        scanf("%s", s);

        if(palindrom(s)==1)
```

```
                printf("Uneta niska je palindrom!\n");
        else
                printf("Uneta niska nije palindrom!\n");

        return 0;

}
```

RESENJE 2

```
#include <stdio.h>
#include <string.h>

#define MAX_DUZINA 20
/* funkcija palindrom vraca vrednost 1 ako je niska s palindrom, inace vraca
vrednost 0 */
int palindrom(char s[]);
int main(){

        char s[MAX_DUZINA+1];
                printf("Unesite nisku: ");
        scanf("%s", s);

        if(palindrom(s)==1)
                printf("Uneta niska je palindrom!\n");
        else
                printf("Uneta niska nije palindrom!\n");

        return 0;

}

int palindrom(char s[])
{
        int n;
        int i;

        n=strlen(s);

        for(i=0;i<n/2;i++)
                if(s[i]!=s[n-1-i])
                        return 0;

        return 1;
}
```

3.

Napisati program koji utvrduje koliko se puta data rec pojavljuje u datoj tekstualnoj datoteci.
Naziv datoteke i rec se zadaju kao argumenti komandne linije.
Pretpostaviti da rec sadrzi manje od 15 karaktera.

```
#include <stdio.h>
#include <stdlib.h>

#define MAX_DUZINA 15

int main(int argc, char* argv[]){
        FILE* ulaz;
        char rec[MAX_DUZINA];
```

```
        int broj_pojavljivanja;

        if(argc!=3){
                fprintf(stderr, "Pogresan broj argumenata!\n");
                exit(EXIT_FAILURE);
        }

        ulaz=fopen(argv[1], "r");
        if(ulaz==NULL){
           fprintf(stderr, "Problem prilikom otvaranja datoteke %s\n", argv[1]);
                exit(EXIT_FAILURE);
        }


        broj_pojavljivanja=0;
        while(fscanf(ulaz, "%s", rec)!=EOF){
                if(strcmp(rec, argv[2])==0)
                        broj_pojavljivanja++;
        }

        printf("Rec %s se pojavljuje %d puta.\n", argv[2], broj_pojavljivanja);
        fclose(ulaz);

        return 0;

}
```

4. U datoteci, cije se ime zadaje preko komandne linije, dat je neki tekst. Napisati program koji sifruje dati tekst tzv. Cezarovom sifrom, i rezultuju´ci tekst upisuje u drugu datoteku, cije ime se takodje zadaje preko komandne linije. Ukoliko druga datoteka nije navedena ispisati rezultat na standardni izlaz.

Izvorno	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z
Šifrovano	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C

ANALIZA

1. Znaci koji nisu slova ostaju neizmenjeni.

2. ASCII šifre svih znakova u opsegu od „A" do „W" treba povećati za 3,

ASCII šifre znakova u opsegu od „X" do „Z" treba umanjiti za 23.

SKICA RESENJA

```
#include <stdio.h>
#include <stdlib.h>

#define MAX_DUZINA 30

void Cezar( char s[],int pom );

int main(int argc, char* argv[]){

        FILE* ulaz, *izlaz;
        char rec[MAX_DUZINA];


        if(argc==2)
```

```
{
    izlaz=stdout;
 }
else
 if(argc==3)
 {
    izlaz=fopen(argv[2], "w");
    if(izlaz==NULL){
        fprintf(stderr, "Problem prilikom otvaranja datoteke %s\n", argv[2]);
        exit(EXIT_FAILURE);
 }
 }
 else if(argc!=3) {
        fprintf(stderr, "Pogresan broj argumenata!\n");
        exit(EXIT_FAILURE);
 }

 ulaz=fopen(argv[1], "r");
 if(ulaz==NULL){
   fprintf(stderr, "Problem prilikom otvaranja datoteke %s\n", argv[1]);
        exit(EXIT_FAILURE);
 }


 while(fscanf(ulaz, "%s", rec)!=EOF){
        Cezar( rec,3 );       /*obrada reci po zadatoj shemi A->D, Z->C */
        fprintf(izlaz, "%s ", rec );

 }


 fclose(ulaz);
 fclose(izlaz);

 return 0;

}
```

RESENJE

```
#include <stdio.h>
#include <stdlib.h>
#define MAX_DUZINA 30
void Cezar( char s[],int pom );

int main(int argc, char* argv[]){
	FILE* ulaz, *izlaz;
	char rec[MAX_DUZINA];

	if(argc==2)
	{
	   izlaz=stdout;
	}
	else
	 if(argc==3)
	{
	   izlaz=fopen(argv[2], "w");
	   if(izlaz==NULL){
	       fprintf(stderr, "Problem prilikom otvaranja datoteke %s\n", argv[2]);
	       exit(EXIT_FAILURE);
	}
	}
	else if(argc!=3) {
	       fprintf(stderr, "Pogresan broj argumenata!\n");
	       exit(EXIT_FAILURE);
	}

	ulaz=fopen(argv[1], "r");
	if(ulaz==NULL){
	  fprintf(stderr, "Problem prilikom otvaranja datoteke %s\n", argv[1]);
	       exit(EXIT_FAILURE);
	}


	while(fscanf(ulaz, "%s", rec)!=EOF){
```

```
            Cezar( rec,3 );       /*obrada reci po zadatoj shemi A->D, Z->C */
            fprintf(izlaz, "%s ", rec );

        }


        fclose(ulaz);
        fclose(izlaz);

        return 0;

}

/* Cezar: sifrira string Cezarovom sifrom, ASCII  */
void Cezar( char s[] ,int pomeraj)
{
  int i;
  for(i=0; s[i]!='\0'; ++i )
  { if  ( s[i]>='A' && s[i]<='Z') s[i]='A' +(pomeraj   +     s[i]-'A')%26;
     else  if  ( s[i]>='a' && s[i]<='z') s[i]='a'+   ( pomeraj +    s[i]-'a') %26;
  }
return;
}
```